Augmenting Conceptual Design Trajectory Tradespace Exploration with Graph Theory

Patrick D. Dees* and Mathew R. Zwack†
Jacobs ESSSA Group, Huntsville, AL, 35806, United States
Michael Steffens‡ and Stephen Edwards§
Georgia Institute of Technology, Aerospace Systems Design Laboratory, Atlanta, GA, 30312, United States

Within conceptual design changes occur rapidly due to a combination of uncertainty and shifting requirements. To stay relevant in this fluid time, trade studies must also be performed rapidly. In order to drive down analysis time while improving the information gained by these studies, surrogate models can be created to represent the complex output of a tool or tools within a specified tradespace. In order to create this model however, a large amount of data must be collected in a short amount of time. By this method, the historical approach of relying on subject matter experts to generate the data required is schedule infeasible. However, by implementing automation and distributed analysis the required data can be generated in a fraction of the time. Previous work focused on setting up a tool called multiPOST capable of orchestrating many simultaneous runs of an analysis tool assessing these automated analyses utilizing heuristics gleaned from the best practices of current subject matter experts. In this update to the previous work, elements of graph theory are included to further drive down analysis time by leveraging data previously gathered. It is shown to outperform the previous method in both time required, and the quantity and quality of data produced.

Nomenclature

ACO	Advanced Concepts Office
DOE	Design of Experiments
MST	Minimally Spanning Tree
POST	Program to Optimize Simulated Trajectories
RMSE	Root Mean Squared Error
RSE	Response Surface Equation
SME	Subject Matter Expert

I. Introduction

Conceptual design is characterized by rapid changes to virtually every element of a system under development. Layout, constraints, requirements, and even objectives are subject to trades. Not only *should* each be investigated, they *must* be to identify those threats and opportunities which are usually only uncovered during later phases of design.[1] Competition between designs and creativity in analysis is important in this critical stage,[2] as no matter how well researched a constraint or requirement is, day-of performance and envi-

*Trajectory Engineer, Advanced Concepts Office, George C. Marshall Space Flight Center/ED04, Huntsville, AL, AIAA Member.
†Systems Engineer, Advanced Concepts Office, George C. Marshall Space Flight Center/ED04, Huntsville, AL, AIAA Member.
‡PhD Candidate, School of Aerospace Engineering, 270 Ferst Drive, Mail Stop 0150, AIAA Student Member.
§Research Engineer, School of Aerospace Engineering, 270 Ferst Drive, Mail Stop 0150, AIAA Member.

ronments can still come as a surprise. Only through full investigation of the system will critical sensitivities and off-nominal performance be identified.

During conceptual design cost committed is high while design knowledge is low. Design freedom is also high, making it an ideal time to change aspects of the system one way or another to investigate trades. Downstream metrics such as reliability, safety, manufacturability, and operations cost will also be impacted by concept down selection during this phase,[3–6] leading to a commitment of up to 80% of the Life Cycle Cost of the system.[3] Therefore, trades during this early phase are tantamount to the success of a system's program, so that design pitfalls can be identified and assuaged early while they are cheap.[1] Without trades on each aspect of the system, critical sensitivities could be overlooked, which is a major driver of analytical error.[7,8] Sensitivities are not necessarily mono-variate either. Combinations of each potential trade to the system must also be considered, as the true value of the system is in its interconnectivity.[9] Here we quickly fall victim to the curse of dimensionality. A balance must be found between time and analysis, as conceptual design studies must be rapid to keep pace with all the disciplinary and institutional changes inevitably occurring in this fluid phase of design.[10,11] In general, conceptual design studies should be on the order of weeks to a month.[10,12] Therefore, a more efficient method of analyzing the potential trades than "try them all" must be employed to satisfactorily characterize the system.[7,8,13] One method for doing so is the creation of a surrogate model.[14]

A surrogate model is a mathematical approximation of a set of data, usually from a computationally expensive analysis code.[15] It utilizes a closed form equation to provide rapid estimates of the output of some detailed analysis. The accuracy by which it represents the tool or analysis is directly related to the number of relevant dimensions and the volume of data it is fit to. Therefore, in order to predict the behavior of a complex system and capture each of the trades which have been thought of and those which have not, a large data set must be produced to construct a surrogate of high accuracy. The surrogate can then provide the conceptual designer with the confidence they need to make decisions on the trades with which they are presented. In some disciplines, the selected tool or analysis runs very quickly or without much fuss so that data acquisition is not an overly laborious task. However, the ascent trajectory of a launch vehicle is not one of those disciplines.

Ascent trajectory optimization is a particularly difficult task during conceptual design due in part to the level of interconnectivity the analysis has with the other disciplines of launch vehicle design. Changes in materials, updated mass estimates, new engine performance data, and many other pieces of data can dramatically effect the optimal path of a launch vehicle to orbit. In addition, the tools by which this analysis is done are historically difficult to work with,[16] adding another dimension of complexity.

In practice, the current method of employing a trajectory subject matter expert (SME) to optimize each case required for a "try them all" approach is infeasible, as generating the number of cases required for creation of a surrogate model likely does not fit within schedule constraints. In order to reduce the time needed to generate the required data, automation has been employed for the collection of data as described in our previous work.[14] Previous efforts, while successful, greatly taxed computational resources and were difficult to fit to a schedule due to stochasticity inherent in the tool. Herein the authors present a method for leveraging previously acquired data for new trade studies in the context of trajectory optimization. By quantifying each aspect of a system, to include ground rules and assumptions and objectives, a new tradespace under consideration can be informed by previous work via similarity. This is achieved by representing the tradespace as a graph, with vehicles and their missions as nodes and trade similarity as edges.

II. Approach

For launch vehicle conceptual design, trajectory optimization is typically performed using one of two tools: Program to Optimize Simulated Trajectories (POST) and Optimal Trajectories By Implicit Simulation (OTIS). These tools are both widely accepted by the launch vehicle design community and utilize direct shooting and direct collocation optimization schemes, respectively.[17] While results from POST and its main competitor OTIS arrive at very similar solutions,[18] POST is used here and in the previous work because it is the tool in use at NASA Marshall Space Flight Center's Advanced Concepts Office[19,20] (ACO).

In the previous work, the creation of input files and assessment of output files was automated and parallelized within a tool called multiPOST so that multiple cases could be analyzed simultaneously. The inputs for a system were divided into two sections: vehicle- and steering-level. In the former are masses, thrusts, specific impulses, orbit parameters and destination C3 values, all of which define the vehicle and

the mission to perform. In the latter are launch azimuth, pitch rates, throttle coefficients, and any other independent variables which POST can adjust to optimize the trajectory, or how to perform the mission. The collection of steering-level inputs within POST is known as the u-vector. Within a tradespace the vehicle-level inputs are selected as a series of cases by Design of Experiments (DOE). In this manner the number of unique cases which must be analyzed is greatly reduced from a "try them all" approach while still gathering enough information to accurately predict the impact of interaction effects. Each one of these cases are then paired with randomly generated steering-level inputs, producing a number of repetitions for a particular case. Each one of these repetitions are then run, and heuristics based upon the best practices of SMEs within ACO assess the resulting output. The repetition may be invalid, in which case it is marked as failed. If the repetition succeeds in moving through each of the phases defined within the input file, additional heuristics will act upon the output optimized u-vector to bring it to a stationary optimal point. For more information on this process the reader is referred to our previous work.[14] The central update to the previous Monte Carlo based method of repetitions is in the generation of steering-level inputs. In this work, we present a method based on graph theory.

A. Graph Method

Graph theory is the study of graphs, mathematical structures representing pairwise relations between objects.[21] A graph is composed of nodes which are connected by edges. A graph can range in connectivity from complete, where each node is connected via an edge to every other node, down to unconnected where there are no edges present. A graph may be bi-directional where there is no difference between the two nodes associated with an edge, or edges can be directional. Edges may be unweighted, or assigned some value to distinguish them from one another.

If each point in the vehicle-level DOE is represented by a node within a graph, and that graph starts off as complete, then the edges between each node can represent the similarity between one vehicle and another via their quantitative attributes. The shorter the "distance" between two vehicles, the more similar their ascent profiles are likely to be. Similar vehicles and their edges then represent a way to "chain" optimized u-vector information from one case to another. If the vehicles are similar enough, the optimized u-vector can be used by POST on the non-converged case. This is due to the fact that POST is a gradient-based optimizer.[22] Consider Figure 1 below. The blue and green curves represent the response space for two similar cases. They could differ by a mass, thrust, specific impulse, or some combination of attributes. After converging, the green case's u-vector is sitting at the bottom of the green well, as shown by the green point. Chaining then uses this optimal u-vector for the blue case, as shown by the gray arrow. Using this u-vector, optimized for a similar vehicle and/or mission, POST will not have to work very hard to bring this initial point along the blue arrow to its optimum.

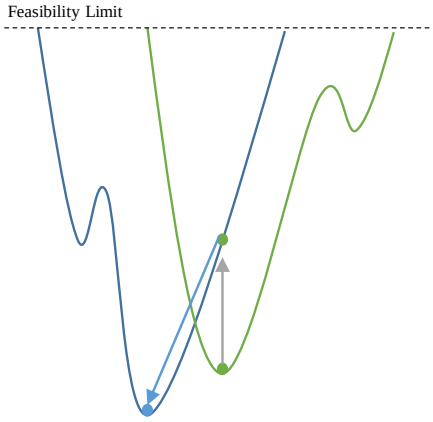

Figure 1: Chaining converged u-vector

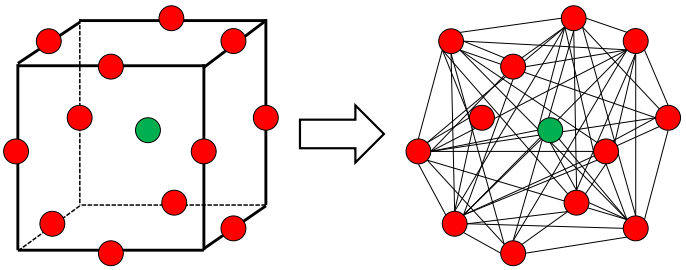

Figure 2: Conversion of cases to complete graph

Each dimension of the vehicle can have wildly different orders of magnitude, and so in order to calculate the distance between any two vehicles in this n-dimensional space the standardized Euclidean distance metric is used. The distance between two points (i, j) is then calculated as

$$d_{ij} = \sqrt{\sum (u_i - v_i)^2 / V[x_i]}$$

where $V[x_i]$ is the variance of dimension i.

The number of edges within a complete graph grows as $n(n-1)/2$, so for a DOE of 500 points, there are almost 125,000 potential connections to try, shown notionally in Figure 2. Therefore, a more efficient method of traversing the graph is desired. This method comes from network theory, a subset of graph theory, which employs combinatorial optimization. In applied mathematics and theoretical computer science, combinatorial optimization focuses on finding an optimal object from a finite set of objects.[23] In this context, the finite set of objects is the set of all connections between nodes representing points of a DOE within a tradespace, and the optimal object is the shortest path which touches each node at least once. One such method to find the shortest path is the minimum spanning tree (MST) algorithm. Many approaches exist[24] but Kruskal's algorithm[25] is used in this work as it is readily available in the Python module NetworkX.[26]

After applying Kruskal's algorithm, the complete graph is reduced to the MST subgraph, shown notionally in Figure 3. The edges remaining in this subgraph are then the shortest distance path to traverse the tradspace. However, a starting point for chaining is still required, shown in Figures 2 and 3 as the green node. The starting point, termed a *seed node*, is produced by analysis before beginning graph based runs. This can result from manual analysis, or from the repetitions method by down-selecting within the DOE. In the latter case, an effective method for choosing these points comes from the shape of the MST. After constructing the MST, certain nodes will have more edges associated with them. The number of edges a node has is termed its degree. Those nodes with the highest degree within the MST will then be the quickest to spread information through the graph by maximizing the number of connections at the onset of analysis. Therefore, by running repetitions on this subset of the DOE and selecting the best performing of the resultant trajectories, the graph method continues to employ the repetitions method as a first step. In the case where a new trade study is being performed which is similar to one already worked, the data from the previous study could be used to seed the graph of the new study.

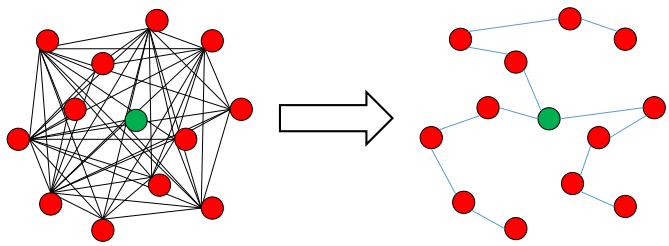

Figure 3: Conversion of complete graph to minimum spanning tree

While creating the MST and chaining optimal u-vectors along the edges does increase the chance that information from a previously optimized case will be acceptable to POST for a case presently under analysis, there are no guarantees. Returning to Figure 1, if the process had been reversed, blue to green, then the chaining would be unsuccessful. This is because the point on the green response curve corresponding to the blue response curve's optimum is beyond the feasibility limit. In the context of POST, the u-vector would not have produced a trajectory which achieved the mission to an acceptable degree by proceeding through each prescribed phase of the ascent. In other cases, a tradespace can include combinations of parameters which are simply not feasible, where no part of the response curve is below the feasibility limit. In yet other cases, sections of the tradespace will have non-linear relationships between vehicle- (DOE) and steering-level (u-vector) inputs, where chaining fails due to high sensitivity to inputs. For each of these examples, chaining may not work immediately. In the first case, chaining could be successful if some intermediary curve was used to "bridge" the blue and green response curves. In the second case, the edge of the feasible region resides somewhere between a converged and unconverged node, which is an important feature of the space to capture. In the third case, feasible regions of the tradespace may remain unexplored without additional help. To address these issues the graph dynamically adds nodes to itself halfway between converged and unconverged nodes when chaining fails. These are referred to as *halfway nodes*.

The process of adding halfway nodes is allowed to occur a finite number of times before halting. It is depicted in Figure 4. Originally, only the green and yellow nodes exist, connected by a black edge. Converged information from the green node is attempting to chain along the edge to the yellow node. However, the yellow node is outside the feasible space for this study. In this context, the yellow node houses some combination of vehicle-level inputs which are not capable of performing the prescribed mission. When chaining to the yellow node fails, the middle red node is created. As this is also outside the feasible region it similarly fails. If the settings allow, another halfway node in purple will then be created, this time luckily just inside the feasible region. Chaining is successful and the resulting converged case data is reported back. After this, the converged u-vector will be chained along to the red node to continue attempting to ultimately reach the yellow node. When this fails the process will end if only two halfway nodes are allowed. By performing halving, the information returned is of a higher granularity than that of the original DOE.

Although the graph method is expected to be more efficient in some arenas of data acquisition, there are others in which it may not. An example of such is cluster computing. Repetitions analysis scales directly with the number of processors available, while the graph method has at all times a finite maximum number of cases that could be analyzed at any given time. This is due to the serial nature by which converged u-vectors are chained along the graph. Although using multiple seeds allows this process to be parallelized, each section of the graph being worked on still has finite connections to work with. However, in the case where a cluster environment is not available, the graph method is expected to outperform repetitions by reducing the amount of time an individual case takes to converge. In addition, since the seed points are assumed to be themselves optimal, cases converged from previously optimized u-vectors require fewer completions to instill confidence in the trajectories analyzed.

B. multiPOST Improvements

Over the year since the previous publication, numerous other improvements have been made to the multiPOST tool. First and foremost, analysis has been distributed across multiple machines, both Linux and

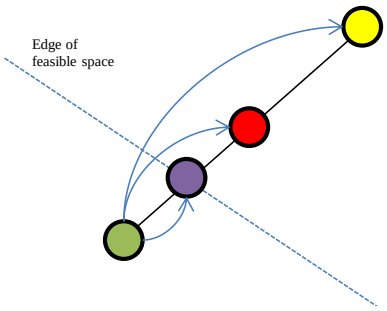

Figure 4: Depiction of halfway nodes capturing feasibility edge

Windows. A single machine runs the algorithms for creating and submitting runs, and watches for converged runs coming back through the network. As these machines have varying performance characteristics it is expected that an individual run will take a varying amount of time to conclude. However, the executables on each system come from identical source code (except for the flags related to OS) and the output data has been verified to be system-agnostic. Second, the distributed machines are outfitted with loggers which enable quick debugging of issues cropping up across the network. Third, the number of rules within the heuristics has increased to recognize a larger number of end states for a completed run. Fourth, many more ways of controlling the creation of input decks and processing outputs have been exposed to the user. In particular, the number of repetitions allocated to a single case has been modified to allow infinite repetitions, normally combined with a requirement on how many converged runs to acquire. Finally, moving from POST3D to the newer POST2 has reduced run times and deck development time.

III. Comparison Studies

An example problem was carried out in order to compare the repetitions-based and graph-based approaches. A single-stage-to-orbit Mars Ascent Vehicle (MAV) was chosen for the example problem due to the simplicity of it's input deck. This inherent simplicity allowed for trades to be performed on vehicle parameters, mission requirements, and objectives.

While on ascent, the vehicle is allowed to throttle each of three engines down to a minimum of 20% at a time optimized upon by POST. The vehicle ascends to an elliptical parking orbit, retaining the delta-V required to achieve a final rendezvous orbit. Insertion into the parking orbit is allowed to range from perigee along the ascending node. Both the propellant and burnout masses are varied, along with the thrust and specific impulse of three identical engines. Launch location is controlled via an initial variable latitude. The apogee and perigee of the parking orbit, along with the delta-V requirement for the final rendezvous orbit are also varied. Atmospheric data used is from NASA Technical Memorandum 82478.[27] The u-vector for the trajectory consists of seven pitch rates, spread over the ascent, the launch azimuth, initial payload, and engine throttling parameter along with the time during ascent to assume the throttle setting. Finally, POST is set to maximize the payload delivered.

A. Parameterization

Parameterization of the MAV comes from a baseline vehicle optimized manually by the author. As this paper is focused on a comparison between methods, exact numbers will only be given on metrics of comparison. Each of the parameters identified above will be varied +/- 10% from baseline values to capture trades on the vehicle design, objectives, and requirements.

Table 1: POST DOE Inputs

Variable	Range
Parking Orbit Perigee	+/- 10%
Parking Orbit Apogee	+/- 10%
Rendezvous Orbit ΔV	+/- 10%
Engine Isp	+/- 10%
Engine Thrust	+/- 10%
Propellant Mass	+/- 10%
Burnout Mass	+/- 10%

B. Study Setup

The comparison studies focus on the two most important aspects of the repetition and graph methods: time to gather the required data for creation of a surrogate model, and the accuracy of the surrogate model created from said data. A major driver of the time required to gather data is the amount of computer power

Table 2: Comparison Trial Settings

Trial	Processors	Required Completions
R1	71	1
R2	71	5
R3	71	10
R4	32	1
R5	32	5
R6	32	10
R7	16	1
R8	16	5
R9	16	10

Trial	Processors	Seed Points
G1	71	10
G2	71	15
G3	71	20
G4	32	10
G5	32	15
G6	32	20
G7	16	10
G8	16	15
G9	16	20

available, and so each method will be run with a varying number of processors allocated to the task.[a] For the repetitions method, the number of required completions for a case is expected to affect the accuracy of the surrogate model, as each converged repetition increases the chance that the globally optimal vehicle has been found. For the graph method, the number of seed points is expected to affect the surrogate similarly. Therefore, surrogate models will be created using data from the repetitions method at varying number of required completions, and from the graph method at varying numbers of seed points. The seed point settings chosen for graph trials represent small percentages of the input DOE (2%, 3%, 4%) where 2% is the minimum typically used for large scale trade studies by the authors. Due to the graph method's capability of producing more cases than the initial DOE by creating halfway nodes, all the data returned by this analysis method will be used for surrogate creation.

Both methods are given the same Latin Hypercube of 500 points from the ranges detailed in Table 1. The Latin Hypercube was used for its ease of creation, as all of the cases within this small tradespace are expected to be feasible.[28,29] For the trials a combination of one Windows laptop, three Windows desktops, and two Linux workstations were used. The POST2 executables used on each of these systems come from identical source code, compiled on their respective operating systems. For the repetitions method, the number of cases in the queue is limited to twice the number of available processors. For the graph method, seeds were taken from the data gained during repetitions.

It should be noted that in the Results section, "Total Time" includes the time to read in input data, start up clients, and perform final results parsing. For the repetitions method there will be a number of runs still within the queue after reaching 100% completion, and in the interest of gathering as much data is possible these are not discarded. The time to analyze those final repetitions is also included.

IV. Results

In both sets of trials, the final product of a surrogate model is the ultimate focus. Therefore, each trial will be compared via how well the data gathered represents the true behavior of the system, quantified by the coefficient of determination, R^2, and Root Mean Squared Error (RMSE) of the surrogate model created from a trial's output data. R^2 ranges from zero to one, and measures how well the model captures the variability of the data. Based upon the authors' experience an R^2 value of 0.999 or better is achievable using repetitions. The second metric is RMSE, which gives a rough estimate of how much error is to be expected for any given value taken from the surrogate. The lower this value is the better the fit is expected to be.

Surrogate model fitting is performed using the statistical software JMP. This software provides tools for fitting many types of surrogates including neural networks, Gaussian process models, and response surface equations. A second order response surface equation (RSE) was selected as the desired fit for trial data

[a]The number 71 was used in lieu of the more predictable 64 because it was the maximum available to the author at the time of publication.

because of its easy construction.

Following the table data for each trial in Figures 9 and 16 are histograms of the percent error distribution resulting from surrogate construction. This is calculated from all the data used to create the surrogate as

$$PercentError = 100 * (Predicted - Actual)/Actual$$

where "Predicted" is the value for the delivered payload coming from the surrogate, and "Actual" is the value returned by POST. In practice, the authors have found that a mean as close to zero as possible, and a standard deviation less than one is desirable. The former requirement tells the analyst that the model has little bias toward over or under predicting, and the latter tells the analyst that the error associated with an arbitrary point is likely less than 1%. For convenience, each histogram is accompanied with the trial signifier it represents, along with the number of processors (associated with "P") and either the number of required completions (associated with "R") or the number of seed points (associated with "S").

Since POST employs a gradient-based (local) optimizer, it has the drawback of finding only local minima; this poses a problem, as previous efforts have shown that the trajectory output space is multimodal.[18,31] Due to this non-convex space it becomes very difficult to determine whether any given data point is the global optimum.[32] It is undesirable to get stuck using suboptimal data, even though POST may have returned it as optimized. For cases with multiple successful completions, the maximum payload delivered is used as the output for the vehicle to more closely replicate the results from an analyst running POST by hand. Many of the successful trajectories that fall below the maximum for a vehicle could easily be adjusted by an expert to achieve a more optimal value. Therefore, they are ignored for surrogate model fitting purposes.

A. Repetitions

For the Repetitions method, trial parameters are the number of processors allocated to the task, and the number of required completions for a case to be considered done. Output metrics of interest are the time required to complete all analyses, the number of repetitions submitted to achieve the desired completions, and surrogate model fit. Trial metrics are shown below in Table 3.

Unsurprisingly, as the number of available processors increase and/or required completions decrease, the time required to gather data tends to decrease, as shown in Figure 5. There is significant variation in simulation time beyond that expected due to processor availability and required completions. This is due to the inherent stochasticity of the repetitions method. An interesting time feature is that as "easier" cases achieve their required number of completions, the added focus on "harder" cases causes the overall rate of case convergence to slow, as shown in Figure 6. The data represented is from Trial R3, where the solid blue line tracks the flow of converged data. The dashed line is a linear fit of this data with an intercept at zero, representing the average rate at which data was returned. For the majority of the trial the rate exceeded the average as the "easy" points of the DOE were optimized. Toward the end of the trial this behavior reversed, going far below the average due to a combination of effects. First, more than the required number of converged points come back for a majority of the points. This is because at the moment when the required

Table 3: Repetition Results

Trial	Time (minutes)	Repetitions Submitted	Successful Repetitions	Surrogate R^2	Surrogate RMSE
R1	301	14,447	1,143	0.961191	897.2031
R2	873	42,707	3,210	0.999622	87.29953
R3	1,520	74,366	5,707	0.999864	52.36811
R4	943	14,561	1,088	0.958935	935.9484
R5	3,049	39,299	3,091	0.999558	94.39713
R6	2,364	71,982	5,613	0.999885	48.13082
R7	797	13,064	1,042	0.952143	1024.412
R8	2,490	39,759	3,039	0.999088	136.0106
R9	4,770	70,984	5,543	0.999897	45.49932

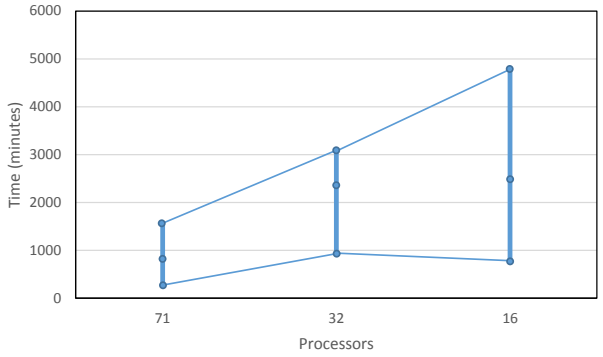

Figure 5: Time required for analyses

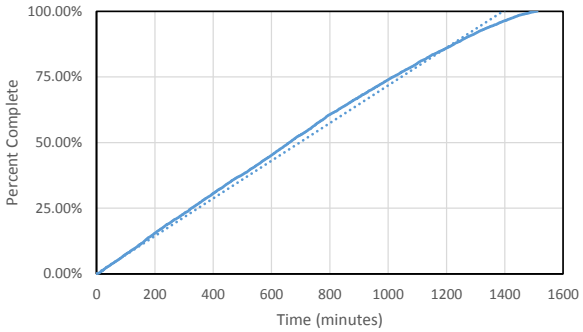

Figure 6: Time history of trial R3

number is reached a few more repetitions of that case may be sitting in the queue. In order to maximize the amount of information gathered, these repetitions are still worked on in the hope of finding the best optima. However, as these are above and beyond the required completions they are not tallied in the plot. In addition, the last few points of the DOE to be converged are likely so due to their particular combination of attributes, constituting a hard vehicle to fly. This could be due to a low thrust-to-weight ratio or low propellant, an unusually small feasible region in the steering-level inputs, or in all possibility a vehicle which cannot achieve the prescribed mission. More repetitions must be submitted then, until random sampling can hit this tiny target. The latter cause is why a maximum number of repetitions is advised.

Another interesting feature is that as available processors are reduced, the number of repetitions submitted tends to reduce as shown in Figure 7. This is due to the queue size creating a bottleneck, having to wait for currently running cases to finish. In this case, there is a reduced chance that more optimal repetitions will be submitted while the required completions are already being worked on. This can be seen in Figure 8 which plots the surrogate model R^2 and RMSE versus the total number of converged data points returned. As expected, as the amount of data available for the surrogate increases, R^2 asymptotically moves toward one and RMSE toward zero. This can also be seen in Figure 9 which shows histograms of the percent error for each trial's surrogate model. Moving down a column in Figure 9 reduces the number of processors, while moving across a row increases the number of required completions. For the one and five required completions trials (first two columns), decreasing the number of processors (moving down a column) results in a mean further from zero and a larger standard deviation. This is consistent with having less data for surrogate construction. However, the ten required set (third column) does not display this behavior, as the higher requirement forces the tool to submit many more repetitions which in turn increases the likelihood of finding a better optima. Increasing the number of required completions improves the surrogate as previously identified, however, here it can be seen that both the mean and standard deviation tend to reduce by orders of magnitude from the additional data present.

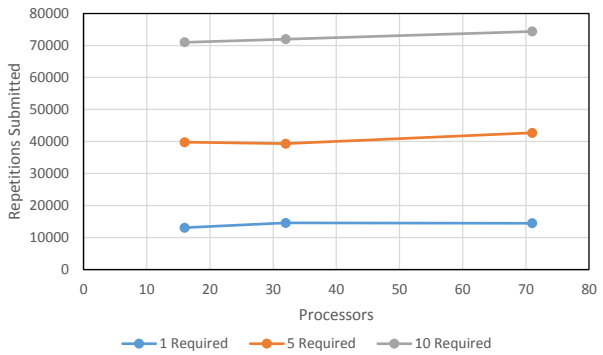

Figure 7: Repetitions submitted versus processors

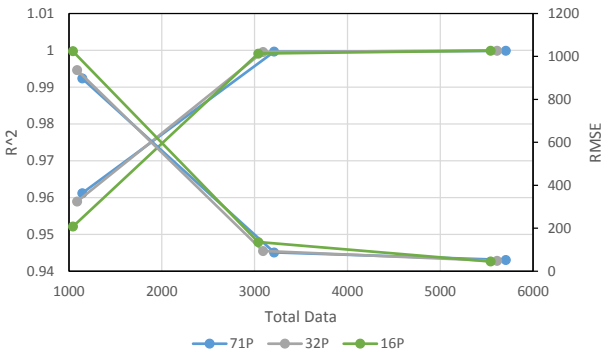

Figure 8: Surrogate fit versus total data

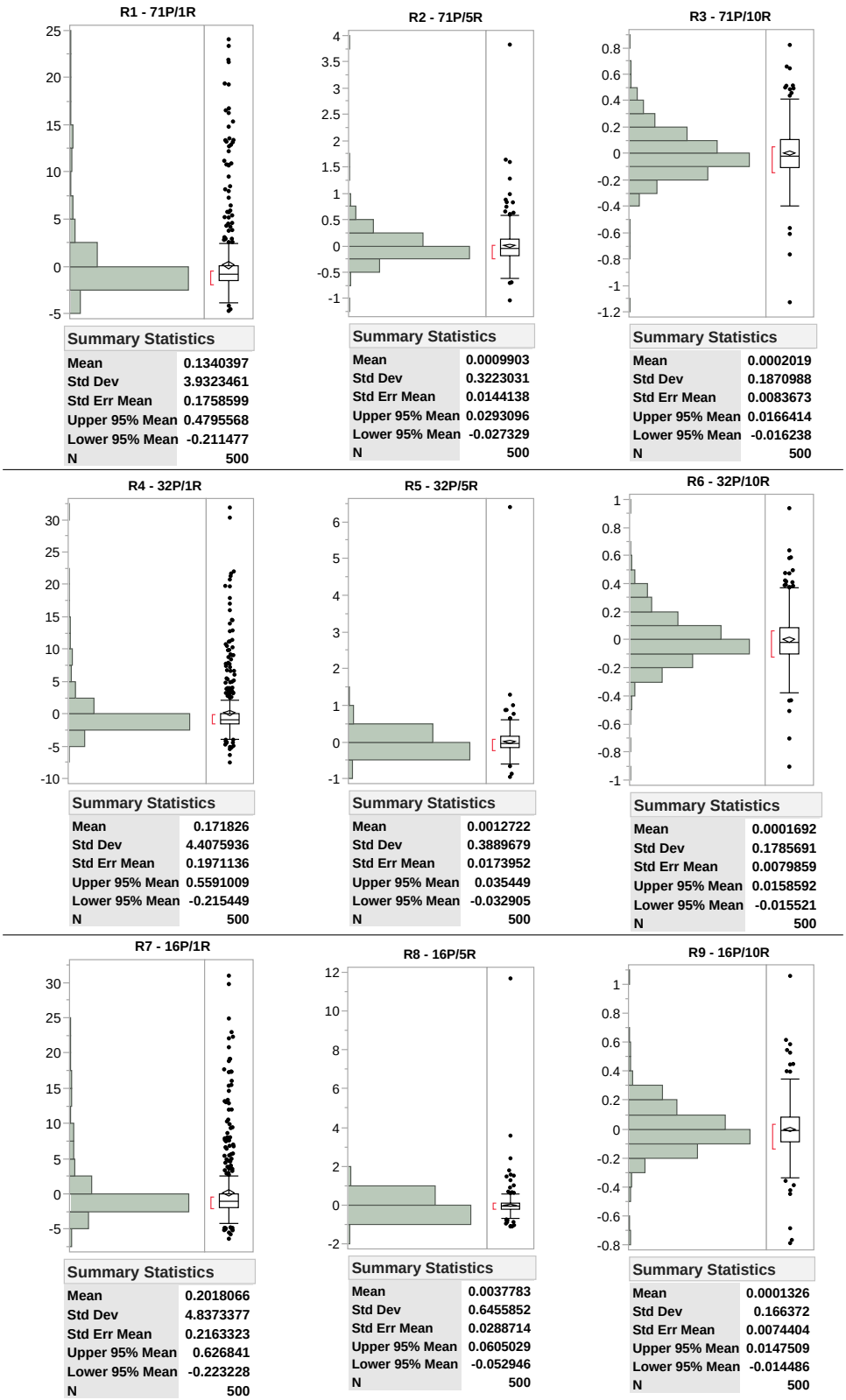

Figure 9: Repetition Trial Histograms of RSE Percent Error

Finally, as Figure 5 shows, the number of allocated processors has a direct effect on the time required to produce the converged runs. Regardless of settings, the number of repetitions submitted which result in returned data hovers around 8%. This small percentage is very much related to the ranges within which the u-vector is sampled, and so it is not a representative number for all studies. However it does show that by increasing the volume of submitted repetitions there will be more data returned, resulting in a higher chance that the globally optimal trajectory will be found for any given vehicle within the DOE. Therefore, the best settings for the repetitions method would then be to maximize allocated processors while also keeping the number of required completions high. While ten completions is an acceptable number here due to the surrogate performance, there is no quantitative measures in use yet for determining when and/or if the true optimal trajectory has been found.

B. Graph

For the Graph method, trial parameters are the number of processors allocated to the task, and the number of seed points. Output metrics of interest are the time required to complete analysis, the percentage of the original DOE submitted which is evaluated, the total number of converged cases which are returned, the total number of data points returned, and the surrogate model fit. Here, evaluated simply means that at least one converged case was reported for that point. Trial metrics are shown below in Table 4.

Table 4: Graph Results

Trial	Time (minutes)	DOE Coverage (%)	Unique Cases	Successful Chainings	Surrogate R^2	Surrogate RMSE
G1	207	77.40	1,109	2,581	0.999978	19.05193
G2	519	83.00	1,179	7,524	0.999980	17.71387
G3	705	80.00	1,392	10,821	0.999982	17.63825
G4	638	85.40	1,371	5,500	0.999977	19.90250
G5	1,341	86.40	1,449	8,185	0.999986	15.60749
G6	1,134	83.80	1,325	10,028	0.999984	16.46271
G7	1,143	81.80	1,325	4,937	0.999983	17.15725
G8	1,428	83.20	1,428	10,576	0.999984	16.16557
G9	2,444	82.80	1,449	13,148	0.999983	16.83919

Unsurprisingly, as shown in Figure 10 below, the number of available processors has an indirect relationship on simulation time. The number of seed points however has a direct relationship. This is due to each seed included representing a different way of flying the vehicle. Over the course of analysis, if a particular seed's trajectory profile is robust then it will successfully chain through a large part of the overall tradespace. Multiply this by the number of seeds and there will be a large number of runs to execute. This is evidenced

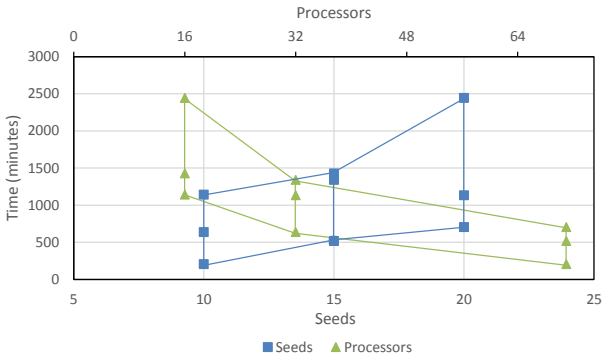

Figure 10: Trial times versus processors and seeds

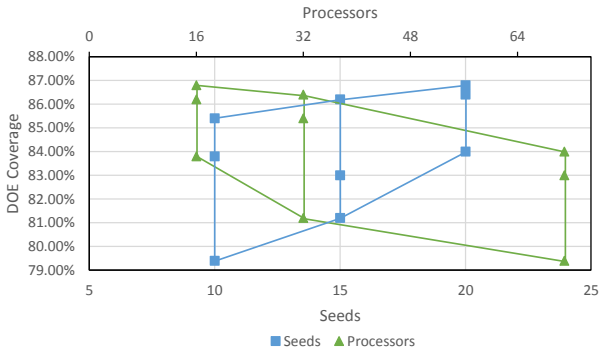

Figure 11: Seeds and processors versus DOE coverage

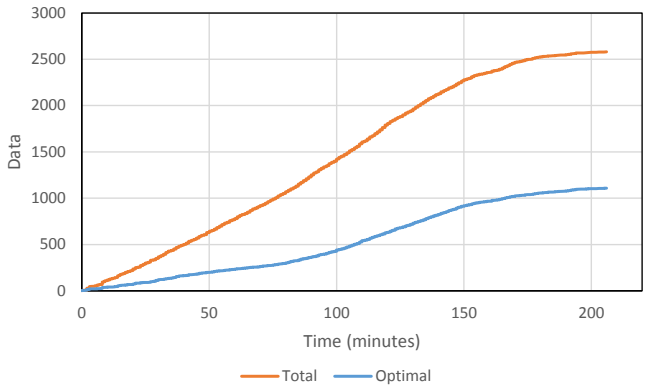

Figure 12: Time history of trial G1

in the "Total Data" column in Table 4, which shows that in several trials the total data returned by the graph method is almost or more than twice that returned by the most prolific repetitions trial.

One of the trends in Figure 11 presents an intriguing relationship. Over the course of graph method trials, increasing the number of seeds increases the likelihood that an arbitrary point from the DOE will be evaluated. This is to be expected, as more information initially present in the graph will increase the chance that at least one seed will chain out to an arbitrary DOE point. However, the figure also depicts an indirect relationship with available processors. This is an unfortunate result of a race condition within the setup of the graph method. It can be thought of as the reverse of the situation in Figure 4. Suppose optimal information is instead coming through from the red node. Chaining directly to the green node in the bottom left-hand corner fails, and so the yellow node is created. In this case the chaining is successful. Chaining once again fails going to the final green node and so the final purple halfway node is created. Finally, chaining fails the third time and due to the simulation settings no more halfway nodes can be created to bridge the gap between yellow and green nodes. This process was due to a single seed attempting to chain through, and so when the next comes along, it may be able to make it to the green node, but not without a halfway node between the red and yellow nodes. If only two halfway nodes are allowed, then the green node is blocked from other seeds. Unfortunately, this behavior cannot be predicted *a priori* and some limit on halving must be in place to avoid wasting time investigating Zeno's Paradox. The authors have found that a setting of five halvings is a fair middle ground.

The time history of a graph trial also shows some interesting trends. Consider the curves in Figure 12, which depict the inflow of converged data from trial G1. In orange is all the data coming in, and in blue is the optimal data ultimately used in the surrogate. The "Total Data" curve is very similar to Figure 6 in that it is fairly linear, bringing in data higher than the average for most of simulation and at the end slowing down. The "Optimal" curve however is far slower. This is a representation of the slow creep of seed data throughout the graph. Each seed has a higher likelihood of being the optimal way to fly for all the points within its initial local neighborhood, but as this chains outward it may no longer be applicable. In this phase a lot more halfway nodes begin to be created, associated with the higher slope region in the middle of the curves. Once this phase abates the rate of data acquisition slows until every possible chaining is enqueued and evaluated.

The relationship between the seed points and simulation are depicted in Figures 13 and 14 for trials G1, G2, and G3. These trials all had the same number of processors allocated (71), and varied the number of seeds (10, 15, 20, respectively). As would be expected, increasing the number of seeds increases the total simulation time and total data returned. The rate at which total data is returned is very similar as the number of processors is constant between these trials, however the rate of optimal data is reduced with increased seed points. This effect is the result of higher competition between the seeds to be the best answer for an arbitrary point in the DOE, and that as more seeds are added they are at lower degree nodes. So while more information is being added, it is of a lower initial efficiency for chaining and thus will take longer to make their way out through the tradespace.

Finally, the surrogate models created from the graph trials were all excellent. While the repetition trials approached having four nines in their R^2 values, *every* graph trial was above 0.9999. In addition, each graph

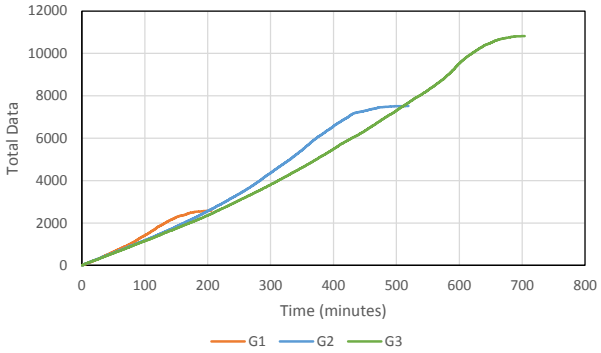

Figure 13: Total data acquisition profile

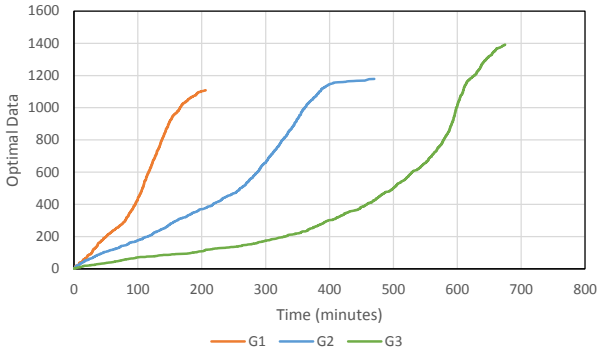

Figure 14: Optimal data acquisition profile

trial's RMSE was less than half the best performing repetitions trial. This is due to the chaining behavior of the graph method. As optimized information is moved about the tradespace, the resulting output is much smoother than what is achieved using Monte Carlo-style sampling. In addition, the creation of halfway nodes increased the amount of data for the surrogate two to three-fold. Comparing the percent error distributions between the repetitions trials in Figures 9 and 16, the mean and standard deviation are in general an order of magnitude lower for the graph trials.

C. Comparison

As was just discussed, the surrogate model created from graph data is of both higher quantity and quality than that coming from repetitions. The repetitions method produces more data points on average for a given case, increasing the likelihood that the global optimum has been found. However, if the seed points used for the graph method are close to or globally optimum, the graph method is more likely to find close to or the global optimum for *more* points within the tradespace via chaining. This is shown primarily in the surrogate metrics columns of Tables 3 and 4. The R^2 and RMSE data for the best and worst trials for each method are depicted in Figure 15. The "Worst Repetitions Trial" columns look strange because their data is actually far off the ranges of the chart. Since the worst repetition trial's R^2 is down near 0.95 and the RMSE over 1,000, it was impossible to place them on the chart and still be able to visually glean information about the other trials depicted. Comparing the "Worst Graph Trial" and the "Best Repetitions Trial", it is easy to see that the graph method far and away outperforms the repetitions method.

Although the repetitions trials can produce acceptable surrogates with enough required completions, the graph trials show that chaining produces a much smoother response space in a far shorter amount of time.

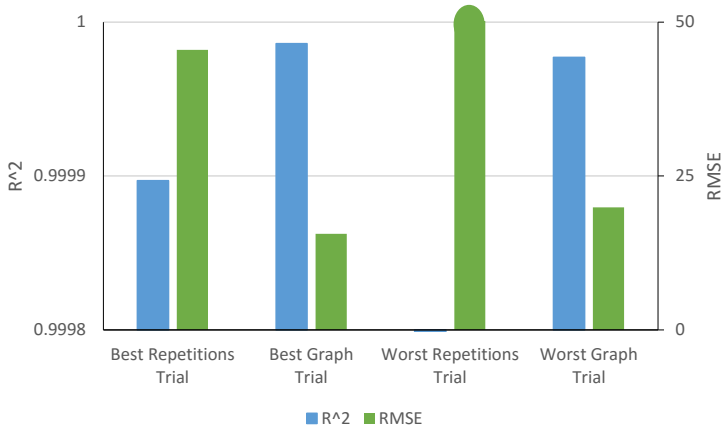

Figure 15: Comparison of best and worst trials

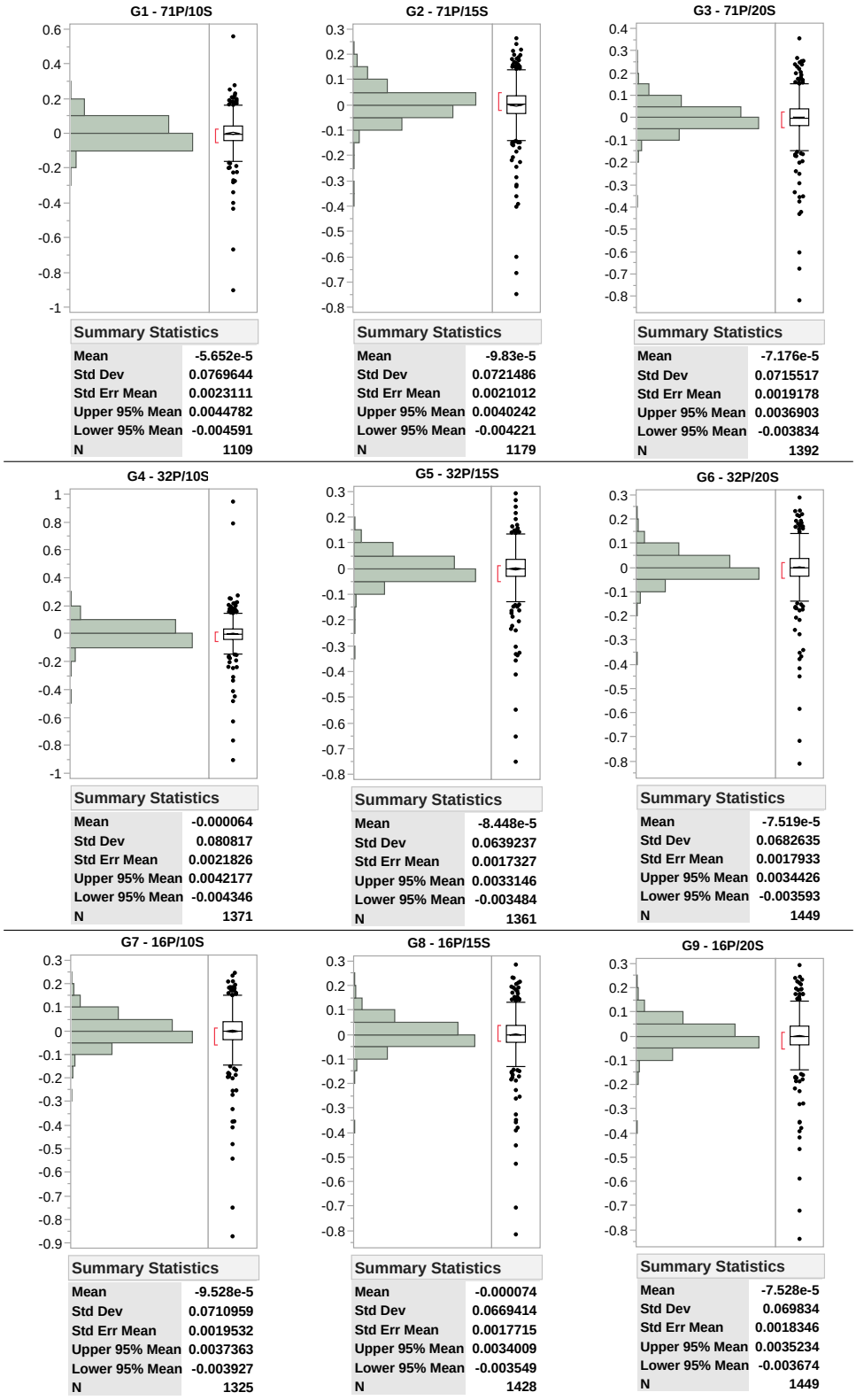

Figure 16: Graph Trial Histograms of RSM Percent Error

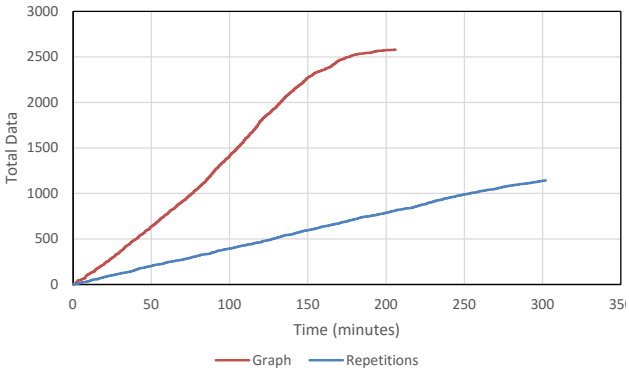

Figure 17: Total data acquisition comparison

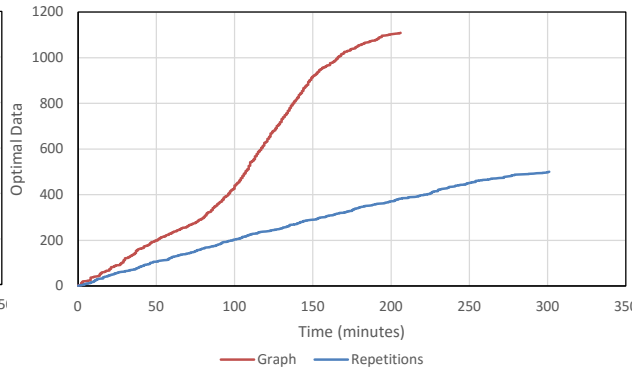

Figure 18: Optimal data acquisition comparison

This can be seen in Figures 17 and 18 which plots the inflow of total and optimal data for the fastest trials, R1 and G1. Slow and steady may win some other race, but that is not the case here. The graph method produced total and optimal data *over 3x faster*. The optimal data produced by the graph method resulted in a surrogate which represented the underlying response space at a far higher degree of accuracy. By both metrics of comparison discussed earlier, the graph method is the clear winner.

V. Conclusion

Each of the methods presented here have their arenas of usefulness. For example, without the repetitions method to supply seed points the user would be right back to running cases manually. For larger case studies with DOEs ranging in the 1,000s of points, optimizing the required seeds would be time prohibitive. The weakness of the repetitions method is not in producing data, but in finding optimal data. Currently, there is no procedure in place for detecting when the best trajectory has been found. If this were included it would be a major boon, as the graph method inherits this weakness when seed points are created via repetitions. In addition, the repetitions method scales up very easily, while the graph method does not. The upper limit for simultaneous analyses for repetitions is far higher than for the graph method, as the graph method has a finite number of executions which can be happening at a given time due to the serial nature of chaining about the tradespace. However, if the repetitions method was let loose in a cluster environment with a far higher number of available processors than the maximum of 71 presented here, there is a distinct possibility that both required time and surrogate accuracy would approach that of the graph method.

The graph method also has the issue of halfway nodes blocking seeds from reaching all corners of the tradespace. Currently there is no fix identified for this issue. However, this behavior automatically puts data at the turning points of the response. Wherever chaining doesn't occur easily is likely a non-linear transition region of the response, the interaction of a critical constraint, or other such important feature of the response space. Having data in these regions is crucial to accurately describing the response space, and is why graph data at its worst outperforms repetitions data.

The final word from the authors is our recommendations for anyone attempting to use these methods. After identifying a tradespace it is recommended that a uniform DOE is created, however the time required may be prohibitive. In this case, a Latin Hypercube is acceptable. After creating the DOE, identify at minimum the cases with the top 2% of degree. Run repetitions on these cases with reasonable maximum per case and required completions settings. Use as many processors as you can get your digital hands on. Once these runs are completed and the optimal information gathered, graph runs need half or less of the computational power used for repetitions. Like many other efforts in life, it is in the cooperation between these two methods that the best results are found.

References

[1]Wertz, J. and Larson, W., *Reducing Space Mission Cost*, Microcosm Press, 1996.

[2]*Marshall Space Flight Center Project Management and Systems Engineering Handbook*, b ed., September 2012, MSFC-HDBK-3173.

[3]Blair, J., Ryan, R., Schutzenhofer, L., and Humphries, W., "Launch Vehicle Design Process: Characterization, Technical Integration, and Lessons Learned," Tech. rep., National Aeronautics and Space Administration, May 2001.

[4]Dulac, N. and Leveson, N., "Incorporating safety risk in early system architecture trade studies," *Journal of Spacecraft and Rockets*, Vol. 46, No. 2, 2009, pp. 430 – 437.

[5]Ullah, R., Zhou, D.-Q., Zhou, P., Hussain, M., and Amjad Sohail, M., "An approach for space launch vehicle conceptual design and multi-attribute evaluation," *Aerospace Science and Technology*, 2011.

[6]Zwack, M., *A Conceptual Reliability Growth Approach for Comparison of Launch Vehicle Architectures*, Ph.D. thesis, Georgia Institute of Technology, Atlanta, GA, December 2014.

[7]Larson, W. and Pranke, L., *Human Spaceflight: Mission Analysis and Design*, McGraw-Hill Companies, 1st ed., October 1999.

[8]Wertz, J., Everett, D., and Puschell, J., *Space Mission Engineering: The New SMAD*, Microcosm Press, 2011.

[9]Rechtin, E., *Systems Architecting of Organizations: Why Eagles Can't Swim*, No. 13 in Systems Engineering, CRC Press, 1st ed., July 1999.

[10]Raymer, D., *Aircraft Design: A Conceptual Approach*, American Institute of Aeronautics and Astronautics, Inc., 4th ed., 2006.

[11]*NASA Systems Engineering Handbook*, National Aeronautics and Space Administration, December 2007, NASA/SP-2007-6105 Rev 1.

[12]Mulqueen, J., Maples, D., and Fabisinski, L., "Tailoring Systems Engineering Processes in a Conceptual Design Environment: A Case study at NASA Marshall Spaceflight Center's ACO," *INCOSE International Symposium*, Rome, Italy, July 2012.

[13]Humble, R., Henry, G., and Larson, W., *Space Propulsion Analysis and Design*, McGraw-Hill Companies, 1st ed., 1995.

[14]Dees, P., Zwack, M., Steffens, M., Edwards, S., Diaz, M., and Holt, J., "An Expert System-Driven Method for Parametric Trajectory Optimization During Conceptual Design," *AIAA SPACE Conference and Exposition*, Pasadena, CA, August 2015.

[15]Simpson, T., Booker, A., Ghosh, D., Giunta, A., Koch, P., and Yang, R., "Approximation Methods in Multidisciplinary Analysis and Optimization: A Panel Discussion," *9th AIAA/ISSMO Symposium on Multidisciplinary Analysis & Optimization*, Atlanta, GA, September 2002.

[16]Steffens, M., *Trajectory-Based Launch Vehicle Performance Analysis for Design-Space Exploration in Conceptual Design*, Ph.D. thesis, Georgia Institute of Technology, August 2016.

[17]Steffens, M., *A Combined Global and Local Methodology for Launch Vehicle Trajectory Design-space Exploration and Optimization*, Master's thesis, Georgia Institute of Technology, 2014.

[18]Nelson, D., *Qualitative and Quantitative Assessment of Optimal Trajectories by Implicit Simulation (OTIS) and Program to Optimize Simulated Trajectories (POST)*, Master's thesis, Georgia Institute of Technology, April 2001.

[19]Waters, E., Garcia, J., Beers, B., Philips, A., Holt, J., and Threet, G., "NASA Advanced Concepts Office, Earth-To-Orbit Team Design Process and Tools," *AIAA SPACE 2013 Conference & Exposition*, 2013.

[20]Zwack, M. R. and Dees, P. D., "Application of Design of Experiments and Surrogate Modeling wiwith the NASA Advanced Concepts Office, Earth-to-Orbit Design Process," 2016.

[21]Bondy, J. and Murty, U., *Graph Theory with Applications*, Elsevier Science Publishing Co., 1982.

[22]Powell, R., Striepe, S., Desai, P., Braun, R., Brauer, G., Cornick, D., Olson, D., Peterson, F., and Stevenson, R., *Program To Optimize Simulated Trajectories (POST) Vol II: Utilization Manual*, 1997.

[23]Schrijver, A., *Combinatorial Optimization: Polyhedra and Efficiency*, Springer-Verlag, 2003.

[24]Baxlamacci, C. and Hindi, K., "Minimum-weight spanning tree algorithms: A survey and empirical study," *Computers & Operations Research*, Vol. 28, No. 8, 2001, pp. 767–785.

[25]Kruskal, J. J., "On the shortest spanning subtree of a graph and the traveling salesman problem," *Proceedings of the American Mathematical Society*, Vol. 7, No. 1, February 1956, pp. 48–50.

[26]NetworkX, "https://networkx.github.io/," .

[27]Smith, R. and West, G., "Space and Planetary Environment Criteria Guidelines for Use in Space Vehicle Development," Tech. rep., NASA, 1982.

[28]Ryan, T. P., *Modern Experimental Design*, Wiley Series in Probability and Statistics, 2007.

[29]SAS Institute Inc, *JMP 11 Design of Experiments Guide.*, SAS Institute Inc, 2013.

[30]*JMP 11 Fitting Linear Models*, SAS Institute Inc., Cary, NC, 2013.

[31]Steffens, M., Edwards, S., and Mavris, D., "Capturing the Global Feasible Design Space for Launch Vehicle Ascent Trajectories," *AIAA SciTech*, Kissimmee, FL, January 2015.

[32]Vanderplaats, G., *Multidiscipline Design Optimization*, 1st ed., 2007.